ALGORITMI E STRUTTURE DI DATI 28/06/2023

1. Tempo disponibile 120 minuti.
2. Non è possibile consultare appunti, slide, libri, persone, siti web, ecc.
3. Scrivere in modo leggibile, su ogni foglio, nome, cognome e numero di matricola.
4. Le soluzioni agli esercizi che richiedono di progettare un algoritmo devono:
   - spiegare a parole l'algoritmo (se utile, anche con l'aiuto di esempi o disegni),
   - fornire e commentare lo pseudo-codice (indicando il significato delle variabili),
   - calcolare la complessità (con tutti i passaggi matematici necessari),
   - se l'esercizio ammette più soluzioni, a soluzioni computazionalmente più efficienti e/o concettualmente più semplici sono assegnati punteggi maggiori.

IMPORTANTE: Risolvere gli esercizi 1–2 e gli esercizi 3–4 su fogli separati. Infatti, al termine, dovrete consegnare gli esercizi 1–2 separatamente dagli esercizi 3–4.

1. Calcolare la complessità $T(n)$ del seguente algoritmo MYSTERY1:

Algorithm 1: MYSTERY1(INT n) $\rightarrow$ INT

```
x = 1
y = 0
for i = 1, ··· , n do
    x = x * 2
    y = y + MYSTERY2(x)
return y

function MYSTERY2(INT n) → INT
if n ≤ 0 then
    return 1
else
    return MYSTERY2(n/4) + n
```

2. Progettare un algoritmo ricorsivo che, date in input due liste concatenate semplici L_1 e L_2, contenenti chiavi intere, rimuova da L_1 tutti gli interi che compaiono anche in L_2 (senza modificare L_2). Ad esempio, se $L_1 = [2, 5, 3, 1, 10, 1]$ e $L_2 = [3, 1]$ alla fine dell'esecuzione avremo $L_1 = [2, 5, 10]$ e $L_2 = [3, 1]$.
3. Progettare un algoritmo che riceve in input un array di numeri (che possono essere sia positivi che negativi) ordinati in modo ~~non decrescente~~ crescente ed un numero positivo K, e che restituisce in output il numero di valori nell'array inclusi nell'intervallo limitato chiuso $[-K, K]$ (ovvero restituisce il numero di elementi x dell'array tali che $-K \leq x \leq K$).
4. Progettare un algoritmo che riceve in input un grafo orientato $G = (V, E)$, un vertice $v \in V$, ed un numero positivo D, e che restituisce in output il numero di vertici a distanza D da v. Si ricorda che la distanza di un vertice v_2 da un vertice v_1 è il numero minimo di archi da attraversare per spostarsi da v_1 a v_2.